Types of Operating system

- Batch Operating System
- Multiprogramming Operating System
- Multitasking Operating System
- Realtime Operating System
- Distributed Operating System
- Clustered Operating System
- Embedded Operating System

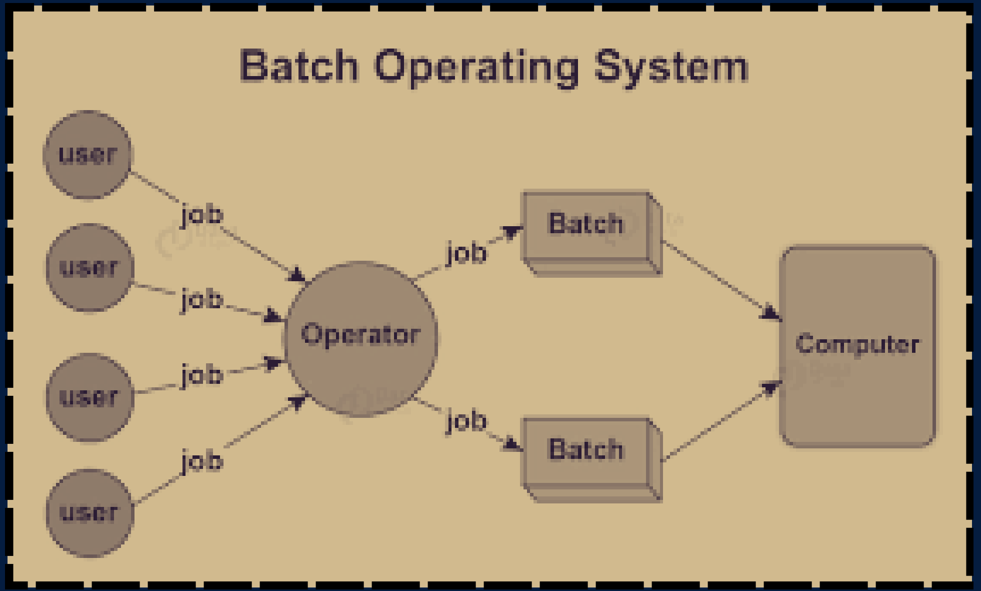
Batch Operating System

Batch operating systems were introduced in the early 1950s to improve efficiency in processing multiple jobs.

Earlier, computers were not interactive devices. Users would prepare a job by writing programs on punch cards or paper tapes, along with input data and instructions.

User-> Job -> Punch cards-> processing-> o/p

In batch processing, these jobs were then submitted to computer operators, who grouped them into batches for sequential processing. The system executed jobs in the order they were received, without user interaction during execution.



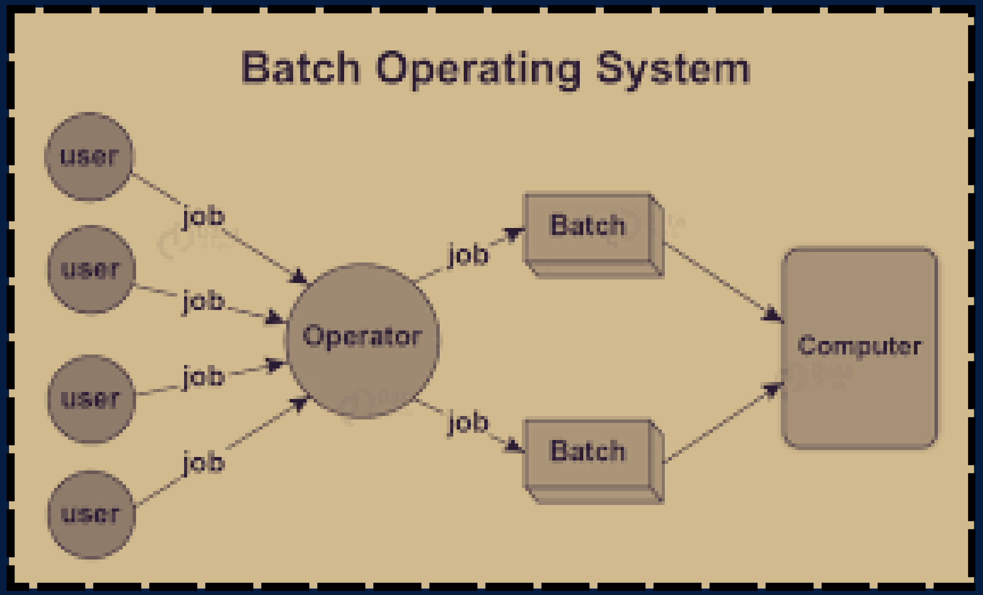
Batch Operating System

A Batch OS groups similar tasks together and processes them without user interaction. It is great for processing large amounts of data that don't need immediate attention, but it takes time to finish the entire batch.

How it works:

- Jobs are created and stored in batches (like a queue).
- The system processes the entire batch one by one without waiting for user input while it runs.
- There's no direct interaction with the user once the job is submitted.

For ex : Early mainframe systems like IBM OS/360, where jobs like payroll processing were done in bulk.

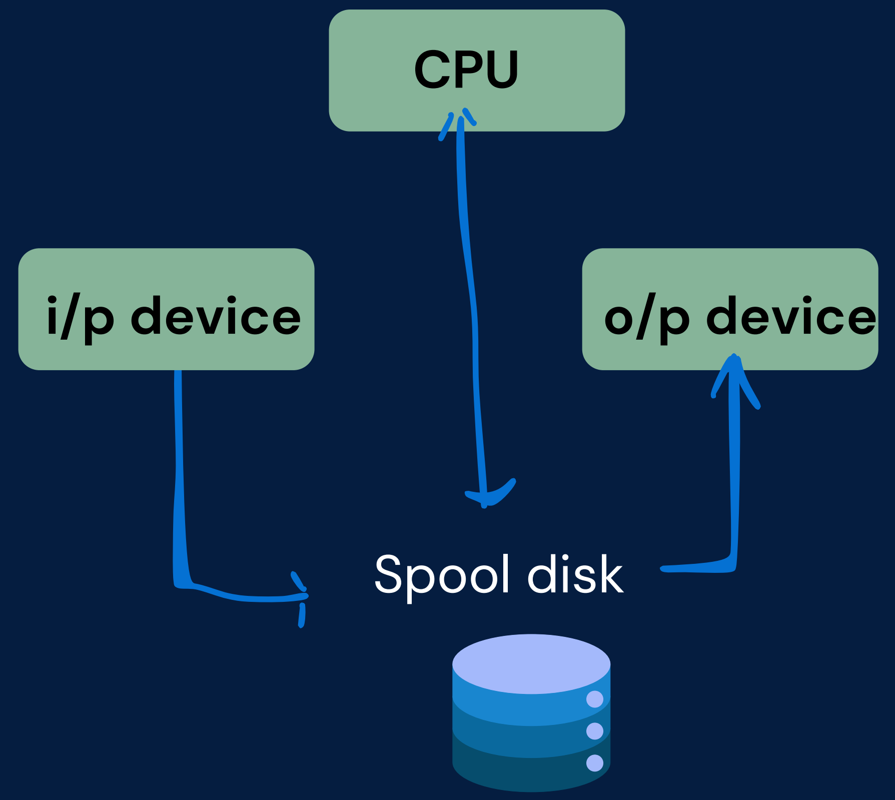


Batch Operating System

In a batch operating system, jobs were executed sequentially without user interaction. However, I/O operations were significantly slower than CPU execution, causing the CPU to remain idle while waiting for input or output.

This inefficiency led to the introduction of spooling, which allowed jobs to be queued in a buffer, enabling the CPU to process other tasks while I/O operations were handled in the background

Spooling is an acronym for simultaneous peripheral operation online. It decouples the CPU from slow I/O devices, allowing batch systems to handle input and output. It created a buffer (spool file on disk) that allowed multiple jobs to be queued.



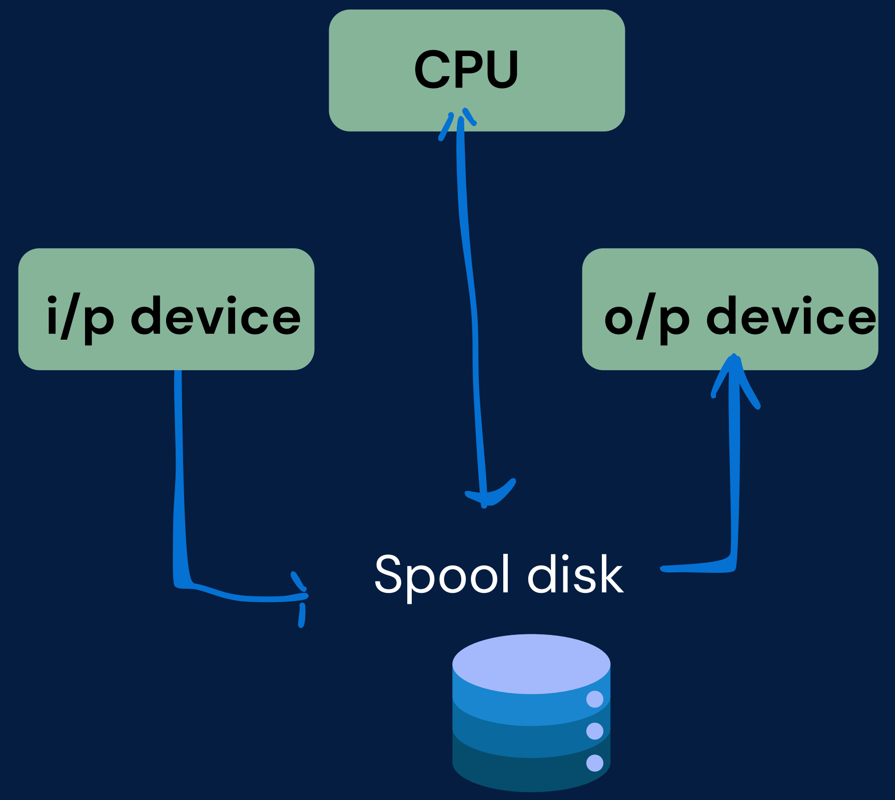
Batch Operating System

Let's say we have three jobs in a batch system:

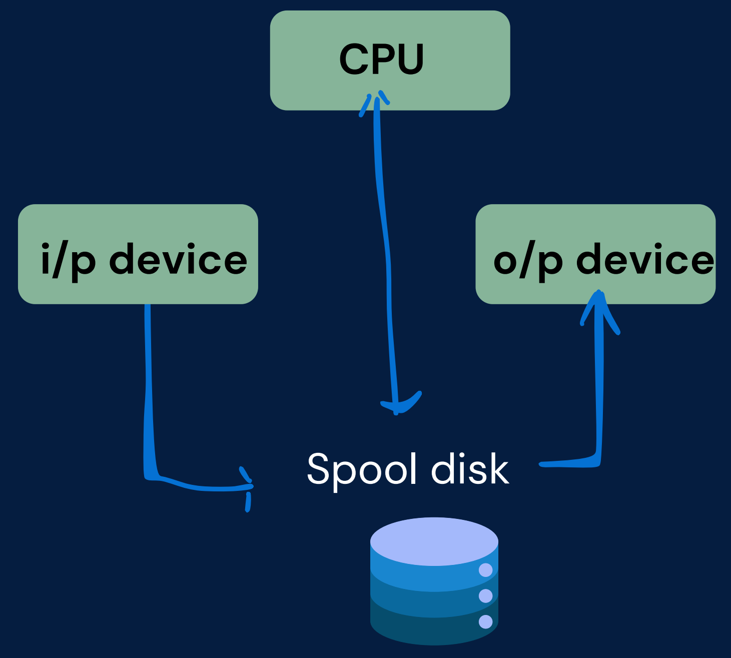
1. Job A → Needs input data
2. Job B → Needs CPU for computation.
3. Job C → Needs output (printer, disk write, etc.).

Without spooling, the CPU would remain idle while waiting for slow input/output operations to complete.

With Spooling → CPU keeps working while input and output happen in the background.



Batch Operating System



```
+------------------+    +------------------+    +------------------+
  |  Job A (Input)  |-----> |   Spool (Disk)  |-----> | Input Device    |
+------------------+    +------------------+    +------------------+
                                    |
                                    v
           +------------------+    +------------------+
             |  Job B (CPU)   |-----> |    CPU Usage   |
           +------------------+    +------------------+
                                    |
                                    v
+------------------+    +------------------+    +------------------+
 |  Job C (Output) |-----> |   Spool (Disk)  |-----> | Output Device   |
+------------------+    +------------------+    +------------------+
```

Step 1: Jobs Enter the Spool (Buffering)

- Job A's input is temporarily stored in a spool (disk buffer).
- Job C's output is also stored in a spool instead of sending it directly to the printer.

Step 2: CPU Utilization is Maximized

- While Job A is waiting for input, the CPU executes Job B (which only needs CPU).
- While Job C is waiting for output to complete, the CPU can process another job instead of being idle.

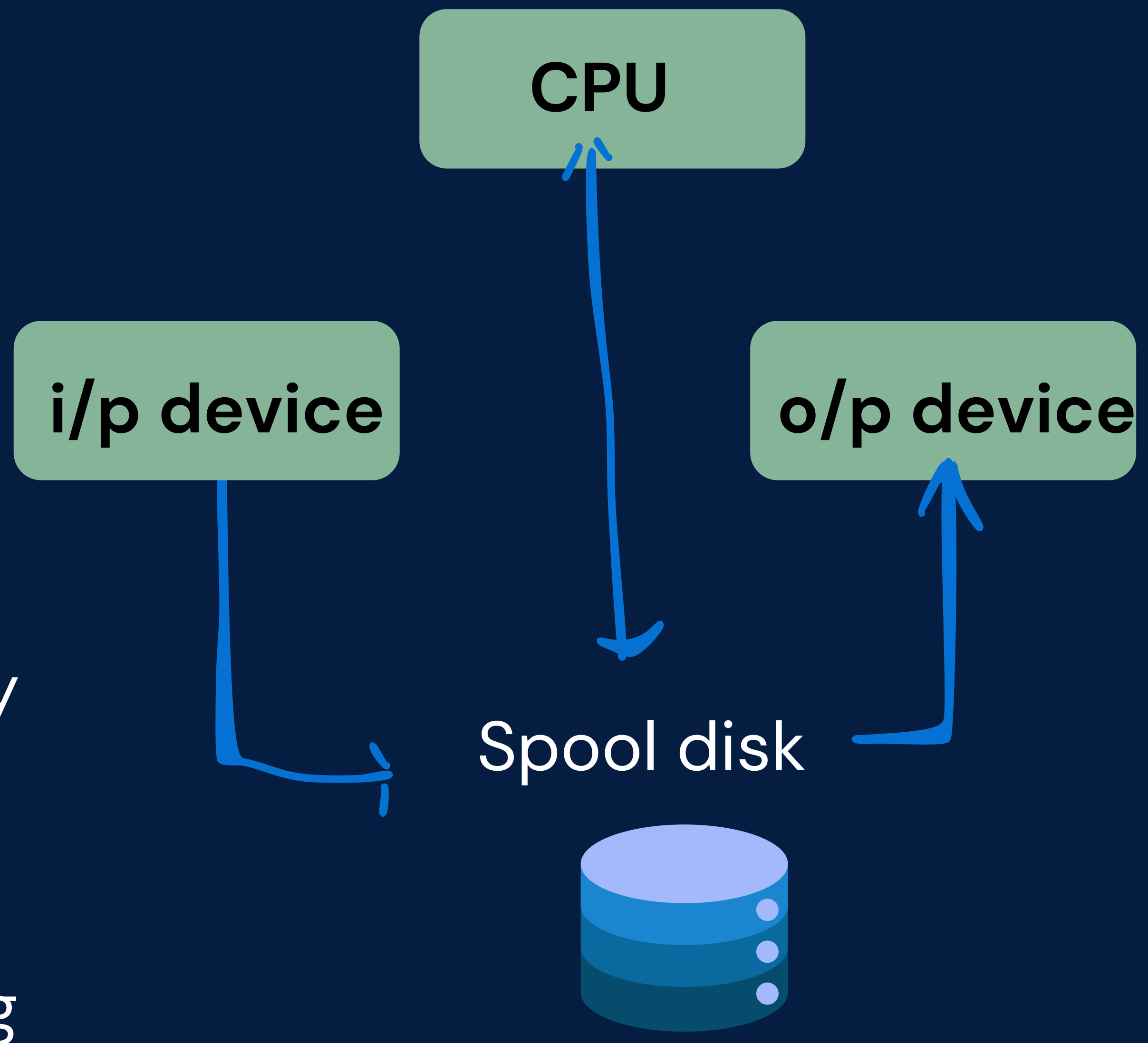
Step 3: Parallel Execution (Overlap of I/O and CPU)

- Job A gets input from the spool instead of directly waiting for the input device.
- Job B executes on the CPU.
- Job C's output is stored in the spool and sent to the printer asynchronously.

Batch Operating System

Steps in Spooling:

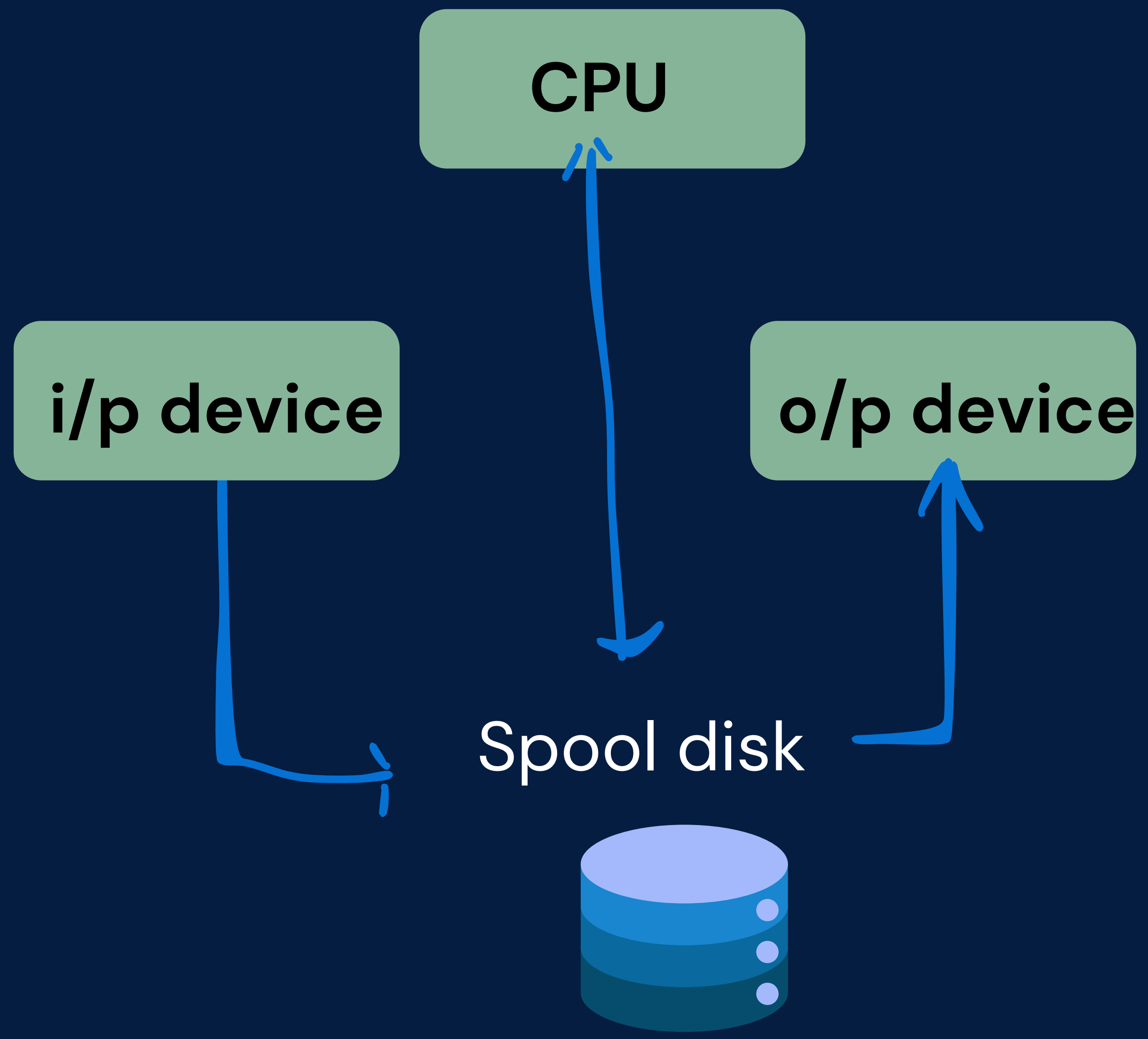
- **Job Submission:** The user submits a job that requires an I/O operation (e.g., printing a document).
- **Temporary Storage (Buffering):** Instead of sending the job directly to the slow I/O device, the OS stores it in a disk buffer (a spool). Multiple jobs can be queued in this buffer.
- **CPU Continues Execution:** While the I/O device is busy processing one job, the CPU continues executing other processes without waiting.
- **I/O Device Processes Jobs from the Queue:** When the I/O device (like a printer) becomes available, it picks the next job from the spool and processes
- **Job Completion & Removal from Buffer:** Once a job is completed, it is removed from the spool, making space for new jobs



Batch Operating System

Printing with Spooling

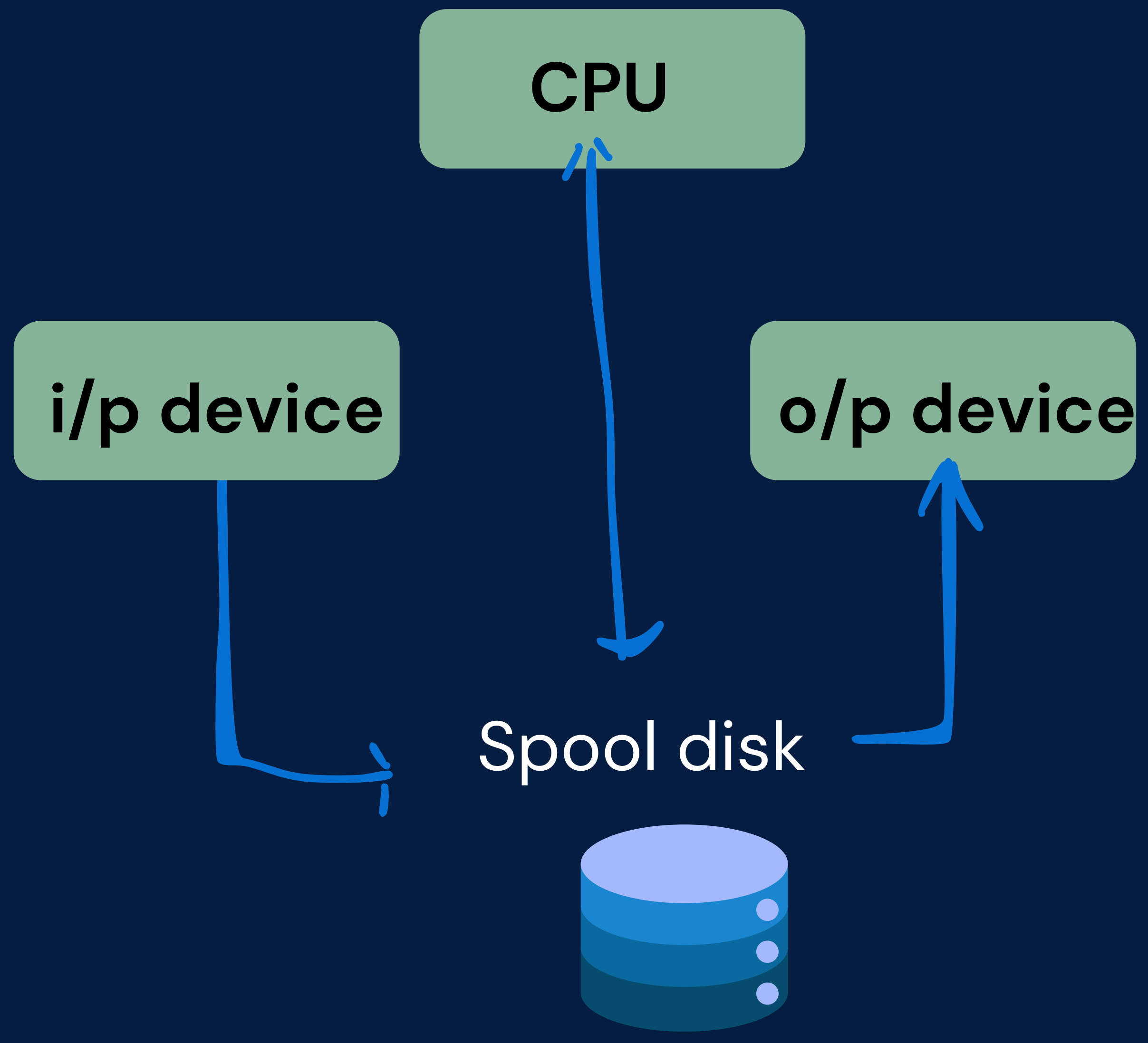
- When you print a document, the OS stores the print job in a queue instead of sending it directly to the printer.
- While the printer slowly prints one document, the CPU continues executing other tasks.
- The printer picks up the next document from the queue when it's free.



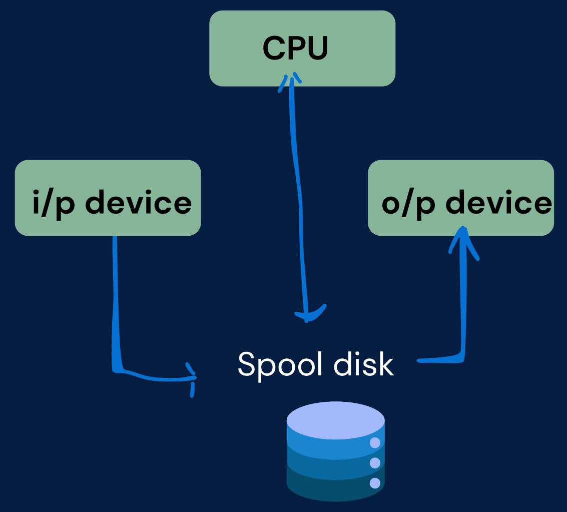
Batch Operating System

Reasons for Spooling:

- **CPU and I/O Speed Mismatch** : CPUs are much faster than I/O devices (like printers or disk drives). Without spooling, the CPU would remain idle while waiting for slow I/O operations to complete.
- **Efficient Resource Utilization** : Spooling allows multiple jobs to be queued in a buffer (usually on disk) so that the CPU can continue processing other tasks while waiting for I/O.
- **Batch Processing Improvement:** In batch systems, spooling helped queue jobs in advance, allowing continuous execution without waiting for manual input.



Batch Operating System



Earlier VS Now after Spooling :

- **Earlier** : Output devices, such as printers, were much slower than the CPU. If a job's output was sent directly to a printer, the CPU would have to wait until the printer finished printing before continuing with the next job
- **Now** : With spooling, the output was written to a disk (a much faster medium) and later printed asynchronously. This allowed the CPU to immediately start processing the next job, significantly improving system throughput.
- **Earlier** : Without spooling, the system could only handle one job's input/output at a time. Input devices like card readers and output devices like printers were idle while the CPU was processing a job.
- **Now** : Spooling allowed the system to overlap I/O and computation While the CPU processed one job, the system could simultaneously read the next job's input from the card reader into a spool and write the current job's output to the spool for later printing.